WHAT IS CLAIMED IS:

1   1.   A method for designing a logic circuit comprising:

2   maintaining a data structure representative of a model,

3   the model including combinational blocks, state elements and

4   graphical library elements of the logic circuit; and

5   generating an architectural model and an implementation

6   model from the data structure.

1   2.   The method of claim 1 wherein the data structure

2   comprises a description of a net list.

1   3.   The method of claim 2 wherein the data structure

2   comprises:

3   elements representing logical functions;

4   elements representing connection points to gates;

5   elements representing all bits of a simulation state; and

6   elements representing an arbitrary collection of bits

7   within the simulation state.

1   4.   The method of claim 4 wherein the elements are all C++

2   classes.

1   5.   The method of claim 1 wherein the architectural model

2   comprises C++ software code.

1   6.   The method of claim 1 wherein the implementation model

2   comprises Hardware Design Language (HDL).

1   7.   The method of claim 6 wherein the HDL is Verilog.

1   8.   The method of claim 6 wherein the HDL is Very high speed

2   integrated circuit Hardware Design Language (VHDL).

1   9.   A method comprising:

2   specifying a model containing combinatorial blocks, state

3   elements and graphical library elements;

4   maintaining a descriptive net list of the model; and

5   generating a C++ model and a Verilog model from the

6   descriptive net list.

1   10.   The method of claim 9 further comprising displaying the

2   model on a graphical user interface (GUI).

1   11.   The method of claim 9 wherein the net list comprises

2   gates, nodes and nets.

1   12.   The method of claim 9 wherein maintaining comprises

2   parsing and analyzing the combinatorial blocks, state elements

3   and graphical library elements of the model.

1   13.   The method of claim 9 wherein generating comprises:

2   partitioning a topology of the net list into a plurality of

3   partitions; and

4   code ordering each of the partitions.

1   14.   A computer program product residing on a computer

2   readable medium having instructions stored thereon which, when

3   executed by the processor, cause the processor to:

4   specify a model containing combinatorial blocks, state

5   elements and graphical library elements;

6   maintain a descriptive net list of the model; and

7   generate a C++ model and a Verilog model from the

8 descriptive net list.

1   15. The computer product of claim 14 wherein the computer

2 readable medium is a random access memory (RAM).

1   16. The computer product of claim 14 wherein the computer

2 readable medium is a read only memory (ROM).

1   17. The computer product of claim 14 wherein the computer

2 readable medium is a hard disk drive.

1   18. A processor and memory configured to:

2   specify a model containing combinatorial blocks, state

3 elements and graphical library elements;

4   maintain a descriptive net list of the model; and

5   generate a C++ model and a Verilog model from the

6 descriptive net list.

1   19. The processor and memory of claim 18 wherein the

2 processor and memory are incorporated into a personal

3 computer.

1   20. The processor and memory of claim 18 wherein the

2 processor and memory are incorporated into a network server

3 residing in the Internet.

1   21. The processor and memory of claim 18 wherein the

2 processor and memory are incorporated into a single board

3 computer.

1   22.   A system comprising:

2   a graphic user interface (GUI) for receiving parameters

3   from a user to generate a model and displaying the model, the

4   model containing combinatorial blocks, state elements and

5   graphical library elements;

6   a maintenance process to manage a data structure

7   representing a descriptive net list of the model; and

8   a code generation process to generate a C++ model and a

9   Verilog model from the data structure.

1   23.   The system of claim 22 wherein the data structure

2   comprises gates, nodes and nets.

1   24.   The system of claim 22 wherein the maintenance process

2   comprises parsing and analyzing the combinatorial blocks,

3   state elements and graphical library elements of the model.

1   25.   The system of claim 22 wherein the code generation

2   process comprises:

3   partitioning a topology of the net list into a plurality

4   of partitions; and

5   code ordering each of the partitions.

1   26.   A data structure comprising:

2   elements representing logical functions of a logic model;

3   elements representing connection points to gates of the

4   logic model;

5   elements representing all bits of a simulation state of

6   the logic model; and

elements representing an arbitrary collection of bits within the simulation state of the logic model.

27. The data structure of claim 26 wherein the elements are stored in a binary tree.

28. The data structure of claim 26 wherein the elements are stored in a linked list.